***Oğuzhan DEMİR 150112825***

***Alihan Küçükgökdemir 150112830***

**CSE238 TERM PROJECT**

**MEMORY GAME**

Firstly, i did it without table and my code is working properly.I didn’t do the bonus part.I used

**INCLUDE Irvine32.inc**

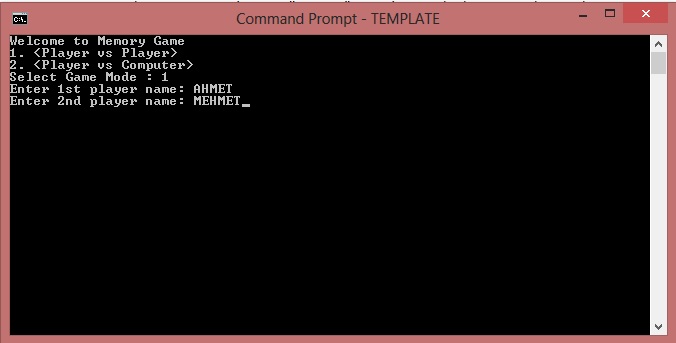
**INCLUDE macros.inc** libraries.I created array that contains 36 character and i created another array size of 36 filled with zeros.This is like “**lock and key**”.If the card is open zero of the choosen coordinate(index) turns to be 1.And this way, program can decide which card should open or not.

Then, i used “**screen**” procedure to show table.By the way, program checks the invalid characters for all time.It choose player to start with “**randclient**” procedure.With “**randarray**” procedure it randomize the characters.While program is working, it keeps the scores of players.If one of the score is greater than 9, it wins and i wrote **“\_restart**” procedure,it asks the user to play another game or exit.This is for ”**player vs player**”.

For “**player vs computer**”, from beginning it asks only one player name.And again it randomize the starter with “**randclient**” proc.Randomize the array with “**randarray**”.Now the difference is, computer’s choice.I wrote another proc called “**randcomp**”.It gives random coordinates within 0-36.And rest of it, is the same as “**player vs player**” side.

My algorithm is like;

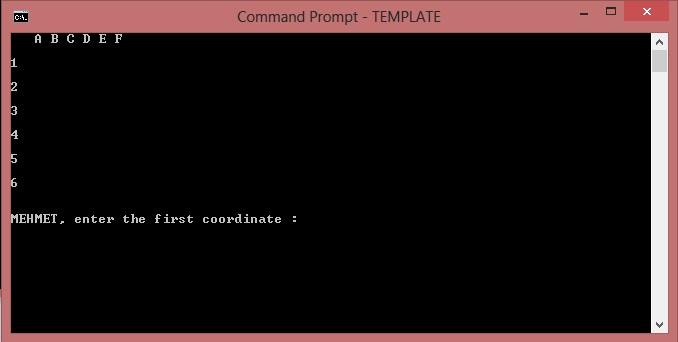
1. Start the program and ask to choose one of the option.And i keep the selection for later using.For ex; if it is “**player vs computer**” code does something else etc.



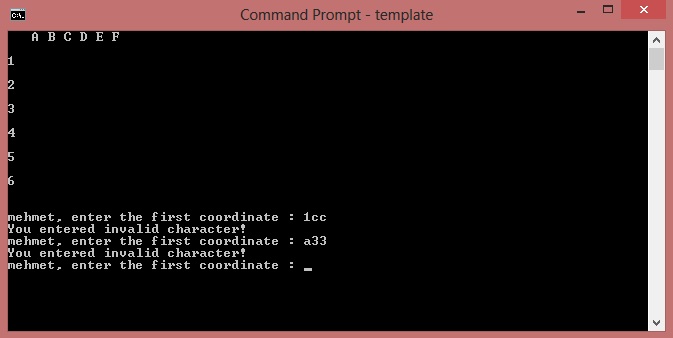
1. Ask the name of the players.If the “**player vs computer**” do not ask the second player.



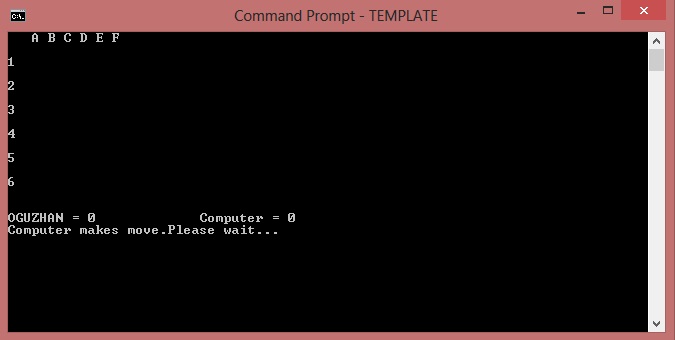
1. Randomize the user and characters with “**randclient**” and “**randarray**” procs.
2. Ask for coordinates for first player and show the characters with “**screen**” proc.



1. If the characters match then increase the score of the player and let the player continue.If not match, skip to the second player.
2. If anytime user enters invalid character give error with sound and written warning.



1. If the game is “**player vs computer**” and if it is computer’s turn ,it gives us random coordinates from “**randcomp**” proc.



1. If one of the score is greater than 9 finish the game and ask the user for another shot.

**PROCEDURES**

**1 ) randclient PROC**

call randomize

mov eax,2

call RandomRange

add eax,1 ; if i dont do this it produces only 0-1.

mov client,eax

ret ;returns the randomized value

**randclient ENDP**

it produces only 1 and 2.

**2 ) randcomp PROC**

mwrite "Computer makes move.Please wait..."

call crlf

mov eax,500

call delay

\_head: ;this equals the letters side A-F

call randomize

mov eax,6

call RandomRange

mov cor1,eax ;randomize 0-5

mov edi,eax

cmp actual[edi],1 ; if opened card jump the head try again

je \_head

\_head2: ; ;this equals the numbers side 1-6

call randomize

mov eax,31

call RandomRange

mov cor2,eax

mov edi,eax

cmp actual[edi],1 ; if opened card jump the head2 try again

je \_head2

cmp eax,cor1 ;if the two coordinates are equal to each other and jump to head

je \_head

mov edi,cor1

mov esi,cor2

mov actual[esi],1 :if there are no problem just open the character “**lock**”

mov actual[edi],1

ret

**randcomp ENDP**

this procedure is for “**player vs computer**” option.

**3 ) randarray PROC**

mov ebx,0

mov edi,0

call randomize

\_start:

mov eax,36

call RandomRange

mov randval,eax

mov esi,randval

mov dl,arraych[esi] ;move character do dl

mov cl,arraych[edi] ;move character do cl

mov arraych[edi],dl ;mov dl to our choosen index

mov arraych[esi],cl ; mov cl to our random index

inc edi

inc ebx

cmp ebx,36

jne \_start

ret

**randarray ENDP**

this procedure is for the randomize our character array.It works like , hold one index and take random index then change them each other.

**4 ) screen PROC**

call clrscr

mwrite "Table is preparing..."

mov eax,500

call delay

call clrscr

mov esi,0

mov edi,0

mov ecx,7

mov al,32

call writechar

l5: ; for the letters

mov al,letter[esi]

call writechar

mov al,32

call writechar

inc esi

loop l5

call crlf

call crlf

mov esi,temp\_esi

mov temp\_esi,0

mov count,sizeof arraych

mov ecx,6

mov edi,0

mov esi,0

l4:

mov temp,ecx

mov al,number[edi] ;for the numbers

call writechar

mov al,32

call writechar

inc edi

mov ecx,6

mov temp\_edi,edi

------------------------------- ----------------------IMPORTANT PART-----------------------------------------------

l3:

mov al,32

call writechar

cmp actual[esi],1 ;check which card is open

jne \_keep ;if not open try other one

mov al,arraych[esi] ;if open mov it to the al and write to the screen

call writechar

jmp \_after

\_keep:

mov temp\_edi,edi

mov al,32

call writechar

\_after: ;check all array

inc esi

loop l3

call crlf

mov ecx,temp

call crlf

loop l4

ret

**screen ENDP**

this procedure is very important.It gives output to the screen.I have 4 array.One for the letters above.One for the numbers side.One for the characters and last one is for the controlling the which character is opened or closed.It works with ,our controller array.It compare the array from beginning to end.If there is opened card it opens .if not it produces “32”(space).

5 ) **\_restart PROC**

call crlf

mwrite "Do you want to play again? (Y/N)"

call readchar

\_h1:

cmp al,89 ;if user enters Y(ASCII)

je \_Y

cmp al,121 ;if user enters X(ASCII)

je \_Y

cmp al,78 ;checks upper or lower case

je \_N

cmp al,110

je \_N

call crlf

mwrite "You entered invalid character!"

call crlf

jmp \_h1

\_Y: ;if game will start again we reset all the variables.

mov rply,1

mov score1,0

mov score2,0

mov score3,0

mov edi,0

mov ecx,36

\_lp1:

mov actual[edi],0

loop \_lp1

jmp \_quit

\_N: ; if not

mov rply,0

call crlf

mwrite "Thanks for playing..."

\_quit:

ret

**\_restart ENDP**

This procedure is useful .It asks to user for another game or finish.